# Manutenção de um sistema de arquivos Linux

www.4linux.com.br

# Sumário



# Índice de tabelas

# Índice de Figuras

# Capítulo 1

# Manutenção de um sistema de arquivos Linux

## 1.1. Objetivos

- Ferramentas e utilitários para manipular e ext2 e ext3;
- Ferramentas e utilitários para manipular reiserfs V3;
- Ferramentas e utilitários para manipular xfs.

## 1.2. Mãos a obra

A manutenção em ambientes GNU/Linux pode ser feita usando diversas ferramentas, dependendo do sistema de arquivos do disco rígido ou partição a ser verificada. O administrador do sistema pode criar, excluir ou verificar partições, afim de manter a vida útil dos discos.

*Mas quais os pacotes que devo instalar para realizar a manutenção?*

Dependendo do sistema de arquivos é preciso instalar pacotes, que trazem ao sistema comandos específicos para cada sistema de arquivos. Vamos a prática:

Para realizar manutenção em sistema de arquivos reiserfs:

```
# aptitude install reiserfsprogs
```

Após a instalação digite o comando baixo para ter a relação de quais comandos é possível utilizar para gerenciar partições reiserfs.

```
# dpkg -L reiserfsprogs | egrep  '(/sbin|/usr/sbin)'
```

```
/sbin
/sbin/mkfs.reiserfs
/sbin/reiserfstune
/sbin/reiserfsck
/sbin/fsck.reiserfs
/sbin/debugreiserfs
/sbin/resize_reiserfs
/sbin/mkreiserfs
```

Vamos repetir os comandos para trabalhar com sistema de arquivos xfs

*# aptitude install xfsprogs*

Após a instalação digite o comando baixo para ter a relação de quais comandos é possível utilizar para gerenciar partições xfs.

*# dpkg -L xfsprogs | egrep '(/sbin|/usr/sbin)'*

```
/sbin
/sbin/xfs_repair
/sbin/mkfs.xfs
/sbin/fsck.xfs
/usr/sbin
/usr/sbin/xfs_logprint
/usr/sbin/xfs_ncheck
/usr/sbin/xfs_rtcp
/usr/sbin/xfs_check
/usr/sbin/xfs_bmap
/usr/sbin/xfs_freeze
/usr/sbin/xfs_growfs
/usr/sbin/xfs_mkfile
/usr/sbin/xfs_quota
/usr/sbin/xfs_metadump
/usr/sbin/xfs_mdrestore
/usr/sbin/xfs_copy
/usr/sbin/xfs_db
/usr/sbin/xfs_info
/usr/sbin/xfs_io
/usr/sbin/xfs_admin
```

Criar sistema de arquivos

Para aplicar um sistema de arquivos em uma partição você pode usar o comando mkfs de 2 maneiras. Vamos a um exemplo prático:

Use o comando mkfs e tecle TAB 2 vezes:

```
mkfs           mkfs.cramfs    mkfs.ext3      mkfs.ext4dev   mkfs.reiserfs
mkfs.bfs       mkfs.ext2      mkfs.ext4      mkfs.minix     mkfs.xfs
```

Veja que em nosso exemplo é possível aplicar sistema de arquivos ext2, ext3, ext4, reiserfs, xfs entre outros.

Para aplicar um sistema de arquivos em uma partição você pode usar o comando mkfs de 2 maneiras, vamos a um exemplo prático.

*# mkfs.ext2 /dev/sda10*

ou

*# mkfs -t ext2 /dev/sda10*

```
mke2fs 1.41.3 (12-Oct-2008)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
124928 inodes, 497980 blocks
24899 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67633152
61 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Seguindo a sintaxe do comando aplique os sistemas de arquivos xfs e reiserfs em outras partições que não estão em uso.

Para XFS:

*# mkfs.xfs /dev/sda11*

```
meta-data=/dev/sda11             isize=256    agcount=4, agsize=31124 blks
         =                       sectsz=512   attr=2
data     =                       bsize=4096   blocks=124495, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096
log      =internal log           bsize=4096   blocks=1200, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=0
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Para Reiserfs:

*# mkfs -t reiserfs /dev/sda12*

```
mkfs.reiserfs 3.6.19 (2003 www.namesys.com)

A pair of credits:
Vladimir Saveliev started as the most junior programmer on the team, and became
the lead programmer.  He is now an experienced highly productive programmer. He
wrote the extent  handling code for Reiser4,  plus parts of  the balancing code
and file write and file read.

Vitaly Fertman wrote  fsck for V3 and  maintains the reiserfsprogs package now.
He wrote librepair,  userspace plugins repair code, fsck for V4,  and worked on
developing libreiser4 and userspace plugins with Umka.


Guessing about desired format.. Kernel 2.6.26-2-686 is running.
Format 3.6 with standard journal
Count of blocks on the device: 124480
Number of blocks consumed by mkreiserfs formatting process: 8215
Blocksize: 4096
Hash function used to sort names: "r5"
Journal Size 8193 blocks (first block 18)
Journal Max transaction length 1024
inode generation number: 0
UUID: 0cca2d8b-d8a5-4a9c-9198-89795a25fa53
ATTENTION: YOU SHOULD REBOOT AFTER FDISK!
        ALL DATA WILL BE LOST ON '/dev/sda12'!
```

Checagem em sistemas de arquivos

Agora vamos entrar em um assunto muito importante que é a checagem do sistema de arquivos. Os comandos que iniciam com **fsck** fazem checagem do sistema de arquivos, e assim como como o **mkfs** você pode encontrar variações, isso vai depender do tipo de sistema de arquivos que seu sistema suporta. Para saber quais sistemas de arquivos você pode checar, digite fsck e tecle TAB 2 vezes:

*# fsck (2x TAB)*

```
fsck          fsck.ext2     fsck.ext4     fsck.minix    fsck.reiserfs
fsck.cramfs   fsck.ext3     fsck.ext4dev  fsck.nfs      fsck.xfs
```

*Não esqueça de antes desmontar as partições para realizar algum tipo de manutenção!*

Para realizar a checagem de um sistema de arquivos em uma partição, você pode usar o comando fsck de 2 maneiras, vamos a um exemplo prático.

*# fsck.ext2 /dev/sda10*

ou

*# fsck -t ext2 /dev/sda10*

```
e2fsck 1.41.3 (12-Oct-2008)
/dev/sda10: clean, 11/124928 files, 18084/497980 blocks
```

Verificar badblocks nas partições:

O comando badblock analisa a partição e identifica setores defeituosos em sistema de arquivos ext2 e ext3. Vamos a exemplo prático:

*# badblock -o badblocks.txt -n -v /dev/sda10*

```
Checking for bad blocks in non-destructive read-write mode
From block 0 to 497982
Testing with random pattern: Pass completed, 0 bad blocks found.
```

Descrição das flags:

-o – Define o nome do arquivo que será uma "lista" com os badblocks encontrados;

-n – Realiza um teste não-destrutivo gravando em cada bloco e depois lendo-o;

-v – Ativa o modo verbose.

Inspeção e alteração em baixo nível

O comando debugfs realiza Inspeção e alteração em baixo nível em sistema de arquivos ext2 e ext3. Vamos a exemplo prático:

debugfs -w <partição>

*# debugfs -w /dev/sda10*

O debugfs abre um prompt interativo onde você pode usar comandos, qua são obtidos através do help:

```
Available debugfs requests:

show_debugfs_params, params
                          Show debugfs parameters
open_filesys, open        Open a filesystem
close_filesys, close      Close the filesystem
feature, features         Set/print superblock features
dirty_filesys, dirty      Mark the filesystem as dirty
init_filesys              Initalize a filesystem (DESTROYS DATA)
show_super_stats, stats   Show superblock statistics
ncheck                    Do inode->name translation
icheck                    Do block->inode translation
change_root_directory, chroot
                          Change root directory
change_working_directory, cd
                          Change working directory
list_directory, ls        List directory
show_inode_info, stat     Show inode information
link, ln                  Create directory link
unlink                    Delete a directory link
mkdir                     Create a directory
rmdir                     Remove a directory
rm                        Remove a file (unlink and kill_file, if appropriate)
kill_file                 Deallocate an inode and its blocks
```

debugreiserfs <partição> - Exibe uma visão geral sobre sistema de arquivos reiserfs. Idem aos comandos dumpe2fs -h e tune2fs -l.

```
# debugreiserfs  /dev/sda12
```

```
debugreiserfs 3.6.19 (2003 www.namesys.com)


Filesystem state: consistent

Reiserfs super block in block 16 on 0x80c of format 3.6 with standard journal
Count of blocks on the device: 124480
Number of bitmaps: 4
Blocksize: 4096
Free blocks (count of blocks - used [journal, bitmaps, data, reserved] blocks):
116265
Root block: 8211
Filesystem is clean
Tree height: 2
Hash function used to sort names: "r5"
Objectid map size 2, max 972
Journal parameters:
        Device [0x0]
        Magic [0x678fa1e5]
        Size 8193 blocks (including 1 for journal header) (first block 18)
        Max transaction length 1024 blocks
        Max batch size 900 blocks
        Max commit age 30
Blocks reserved by journal: 0
Fs state field: 0x0:
```

Exibir informações sobe sistema de arquivos

O comando dumpe2fs pode ser usado para exibir informações do sistema de arquivos como o tipo do sistema de arquivos, características especiais, número de inodes, blocos livres, tamanho do bloco e intervalo entre checagens automáticas. Vamos a exemplo prático:

dumpe2fs -f <partição> – Exibe informações de baixo nível sobre sistema de arquivos ext2 e ext3.

*# dumpe2fs -f /dev/sda10*

```
Filesystem volume name:   <none>
Last mounted on:          <not available>
Filesystem UUID:          69ba3021-da98-4fc6-b2c2-7e5a6ee8322c
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      ext_attr resize_inode dir_index filetype sparse_super
Filesystem flags:         signed_directory_hash
Default mount options:    (none)
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              124928
Block count:              497980
Reserved block count:     24899
Free blocks:              479896
Free inodes:              124917
First block:              1
Block size:               1024
Fragment size:            1024
Reserved GDT blocks:      256
Blocks per group:         8192
Fragments per group:      8192
Inodes per group:         2048
Inode blocks per group:   256
```

dumpe2fs -h <partição> – Exibe uma visão geral sobre sistema de arquivos.

*# dumpe2fs -h /dev/sda12*

Ajustar paramentos em sistema de arquivos

O comando tune2fs pode ser utilizado em partições ext2, ext3 e ext4 para realizar ajustes, como por exemplo converter partições ext2 para ext3 e vice versa, exibir informações de um sistema de arquivos e ainda configurar o tempo para realizar checagem automática feita por fsck na partição.

Opções do tune2fs:

tune2fs -c 30 <partição> - Determina o numero máximo de montagens para realizar checagem automática feita por fsck. Se for 0 (zero) desativa a checagem.

```
# tune2fs -c 30 /dev/sda10
```

tune2fs -i 1m <partição> - Determina o numero máximo de tempo para realizar checagem automática feita por fsck. Se for 0 (zero) desativa a checagem. Use d para dias , m para meses e w para semanas.

```
# tune2fs -i 1m /dev/sda10
```

tune2fs -j <partição> - Atribui o recurso journalling a uma partição ext2.

```
# tune2fs -j  /dev/sda10
```

Para realizar ajustes em sistema de arquivos reiserfs: reiserfstune <partição>

reiserfstune –u UUID <partição> – Altera o UUID da partição.

reiserfstune –l nome <partição> – Altera o Label da partição.

# Capítulo 2

# Gerenciando

## 2.1. Objetivos

- Trobleshooting: Xfs_info, xfs_check e xfs_repair.

## 2.2 Troubleshooting

*Como posso gerenciar sistema de arquivos do tipo XFS?*

Para trabalhar com sistema de arquivos XFS, primeito instale o pacote xfsprogs e use o comando abaixo:

*# dpkg -L xfsprogs | egrep '(/sbin|/usr/sbin)'*

```
/sbin
/sbin/xfs_repair
/sbin/mkfs.xfs
/sbin/fsck.xfs
/usr/sbin
/usr/sbin/xfs_logprint
/usr/sbin/xfs_ncheck
/usr/sbin/xfs_rtcp
/usr/sbin/xfs_check
/usr/sbin/xfs_bmap
/usr/sbin/xfs_freeze
/usr/sbin/xfs_growfs
/usr/sbin/xfs_mkfile
/usr/sbin/xfs_quota
/usr/sbin/xfs_metadump
/usr/sbin/xfs_mdrestore
/usr/sbin/xfs_copy
/usr/sbin/xfs_db
/usr/sbin/xfs_info
/usr/sbin/xfs_io
/usr/sbin/xfs_admin
```

Vamos ver na prática alguns comandos uteis:

xfs_check <partição> - Verifica a consistência no sistema de arquivos xfs.

*# xfs_check /dev/sda13*

```
xfs_check: unexpected XFS SB magic number 0x00000000
xfs_check: read failed: Invalid argument
xfs_check: data size check failed
cache_node_purge: refcount was 1, not zero (node=0x919c670)
xfs_check: cannot read root inode (22)
bad superblock magic number 0, giving up
```

xfs_repair <partição> - Repara erros no sistema de arquivos xfs.

*# xfs_repair /dev/sda13*

```
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
Phase 5 - rebuild AG headers and trees...
        - reset superblock...
Phase 6 - check inode connectivity...
        - resetting contents of realtime bitmap and summary inodes
        - traversing filesystem ...
```

xfs_info <ponto de montagem> - Traz informações sobre tamanho e numero de blocos da partição do tipo xfs.

1 – Crie um ponto de montagem:

*# mkdir /media/backup*

2 – Realize a montagem:

*# mount -t xfs /dev/sda13 /media/backup*

3 – Exiba informações com o comando xfs_info

*# xfs_info /media/backup*

```
meta-data=/dev/sda13             isize=256    agcount=4, agsize=31124 blks
         =                       sectsz=512   attr=2
data     =                       bsize=4096   blocks=124495, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096
log      =internal               bsize=4096   blocks=1200, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=0
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Para gerenciar partições swap utilize os seguintes comandos:

mkswap <partição> - Aplica swap a uma partição.

*# mkswap /dev/sda14*

swapon <partição> - Ativa a partição swap.

```
# swapon /dev/sda14
```

swapon -s – Lista as partições swap ativas.

```
# swapon -s
```

swapoff <partição> - Desativa a partição swap.

```
# swapoff  /dev/sda14
```